[illegible]

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/705,678	11/02/2000	Darrell D. Boggs	042390.P9577	6275

7590 03/09/2004

Eric S Hyman
Blakely Sokoloff Taylor & Zafman LLP
12400 Wilshire Boulevard 7th Floor
Los Angeles, CA 90025

EXAMINER	
HUISMAN, DAVID J	
ART UNIT	**PAPER NUMBER**
2183	7

DATE MAILED: 03/09/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

[illegible]

Offic Action Summary	**Application N .** 09/705,678	**Applicant(s)** BOGGS ET AL.	
	Examiner David J. Huisman	**Art Unit** 2183	

Sv

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *24 February 2004*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-19* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-19* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *24 February 2004* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.

4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.

5) ☐ Notice of Informal Patent Application (PTO-152)

6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-19 have been examined.

Papers Submitted

2. It is hereby acknowledged that the following papers have been received and placed of record in the file: #5. Amendment "A" as received on 2/24/2004 and #6. Formal Drawings as received on 2/24/2004.

Specification

3. The title of the invention is not descriptive. A new title is required that is clearly indicative of the invention to which the claims are directed.

Withdrawn Rejections

4. Through amendments, applicant has overcome the rejections set forth in the Office Action mailed on November 20, 2003, for claims 1-19. However, a new ground(s) of rejection is made below.

Claim Rejections - 35 USC § 102

5. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

(e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

6. Claims 1-4, 6-15, and 17-18 are rejected under 35 U.S.C. 102(b) as being anticipated by Sager, U.S. Patent No. 5,966,544.

7. Referring to claim 1, Sager has taught a processor comprising:

a) a replay queue to receive a plurality of instructions. See the buffer in Fig.7 and column 9, line 50, to column 10, line 2.

b) an execution unit to execute the plurality of instructions. See Fig.7 and column 8, lines 64-67.

c) a scheduler coupled between the replay queue and the execution unit to speculatively schedule instructions for execution. See Fig.7 and column 10, lines 7-32, and note that the mux can be interpreted as a scheduler since it selects one of multiple instructions to send to the execution core. It can be seen that the output instruction of the buffer (replay queue) goes to the mux, wherein the mux will eventually send that instruction to the execution unit.

d) a checker coupled to the execution unit to determine whether each instruction of the plurality of instructions has executed successfully, and coupled to the replay queue to dispatch to the replay queue each instruction that has not executed successfully. See Fig.7 and column 9, lines 50-53.

e) independent instructions and associated dependent instructions are moved to the replay queue upon unsuccessful execution of an independent instruction until data required for successful execution of the independent instruction is valid. See column 9, lines 31-36, column 10, lines 47-52, and column 12, lines 4-8. From these passages, Sager has made it clear that when an independent instruction needs to be replayed, its dependent instructions also need to be replayed.

8. Referring to claim 2, Sager has taught a processor as described in claim 1. Sager has further taught an allocator/renamer coupled to the replay queue to allocate and rename those of a plurality of resources needed by the instruction. See the renamer component in Fig.7 and column 8, lines 33-50.

9. Referring to claim 3, Sager has taught a processor as described in claim 2. Sager has further taught a front end coupled to the allocator/renamer to provide the plurality of instructions to the allocator/renamer. See Fig.7 and column 8, lines 29-32.

10. Referring to claim 4, Sager has taught a processor as described in claim 2. Sager has further taught a retire unit to retire, the plurality of instructions, coupled to the checker to receive those of the plurality of instructions that have executed successfully, and coupled to the allocator/renamer to communicate a de-allocate signal to the allocator/renamer. See Fig.7, Fig.8, and column 14, lines 12-20. Also, it is inherent that when an instruction is retired, its resources (i.e. registers) are deallocated so that another instruction has the option to use them (as opposed to resources continuing to be allocated to an instruction which no longer requires them because that instruction has completed). If these resources were not deallocated, then they would not be available to the processor, thereby inhibiting execution.

11. Referring to claim 6, Sager has taught a processor as described in claim 1. Sager has further taught:

a) at least one cache system on a die of the processor. See the instruction cache and data cache in Fig.7. Also, see Fig.3 and column 7, lines 25-26.

b) a plurality of external memory devices. See Fig.2 and note the use of external main memory and hard disk storage.

c) a memory request controller coupled to the execution unit to obtain a plurality of data from the at least one cache system and the plurality of external memory devices and to provide the plurality of data to the execution unit. See Fig.6 and note that data can be provided from the data cache 310 to the ALU functional unit 300. It is inherent that if data is to be supplied to the execution units, then it must first be retrieved.

12. Referring to claim 7, Sager has taught a processor as described in claim 6. Sager has further taught that the at least one cache system comprises a first level cache system and a second level cache system. See Fig.2.

13. Referring to claim 8, Sager has taught a processor as described in claim 6. Sager has further taught that the external memory devices comprise at least one of a third level cache system, a main memory, and a disk memory. See Fig.2.

14. Referring to claim 9, Sager has taught a processor as described in claim 1. Sager has further taught a staging queue coupled between the checker and the scheduler. See the delay component in Fig.7. Also, see column 36-46, and note that the delay element acts as a queue in that it holds a copy of an instruction for multiple clock cycles until the same instruction completes all stages of execution.

15. Referring to claim 10, Sager has taught a processor as described in claim 1. Sager has further taught that the checker comprises a scoreboard to maintain a status of a plurality of resources. See Fig.8 and column 13, lines 18-31.

16. Referring to claim 11, Sager has taught a processor comprising:

a) a replay queue to receive a plurality of instructions. See the buffer in Fig.7 and column 9, line 50, to column 10, line 2.

b) at least two execution units to execute the plurality of instructions. See Fig.7 and Fig.5, and note the multiple execution cores.

c) at least two schedulers coupled between the replay queue and the execution units to schedule instructions for execution based on data dependencies and instruction latencies. See the mux and TLB/TAG components in Fig.7. From column 10, lines 7-32, note that the mux can be interpreted as a scheduler since it selects one of multiple instructions to send to the execution core. It can be seen that the output instruction of the buffer (replay queue) goes to the mux, wherein the mux will eventually send that instruction to the execution unit. In addition, from column 10, lines 15-25, note that the TLB/TAG component sends a scheduled "fake" instruction to the mux. From Fig.7, it should also be realized that the TLB/TAG component is between the replay queue (buffer) and execution units.

d) a checker coupled to the execution units to determine whether each instruction has executed successfully, and coupled to the replay queue to communicate each instruction that has not executed successfully. See Fig.7 and column 9, lines 50-53.

e) independent instructions and associated dependent instructions are moved to the replay queue upon unsuccessful execution of an independent instruction until data required for successful execution of the independent instruction is valid. See column 9, lines 31-36, column 10, lines 47-52, and column 12, lines 4-8. From these passages, Sager has made it clear that when an independent instruction needs to be replayed, its dependent instructions also need to be replayed.

17. Referring to claim 12, Sager has taught a processor as described in claim 11. Sager has further taught a plurality of memory devices coupled to the execution units such that the checker determines whether the instruction has executed successfully based on a plurality of information

provided by the memory devices. See Fig.8 and column 13, lines 18-31, and note that the scoreboard is a memory device that allows the checker to determine whether an instruction's execution was successful. The scoreboard's operation per cycle is dependent on the instruction that has been executed, which has been provided by the instruction cache (Fig.7) and accesses a register file (Fig.6).

18. Referring to claim 13, Sager has taught a processor as described in claim 12. Sager has further taught an allocator/renamer coupled to the replay queue to allocate and rename those of a plurality of resources needed by the plurality of instructions. See the renamer component in Fig.7 and column 8, lines 33-50.

19. Referring to claim 14, Sager has taught a processor as described in claim 13. Sager has further taught a front end coupled to the allocator/renamer to provide the plurality of instructions to the allocator/renamer. See Fig.7 and column 8, lines 29-32.

20. Referring to claim 15, Sager has taught a processor as described in claim 13. Sager has further taught a retire unit to retire the plurality of instructions, coupled to the checker to receive those of the plurality of instructions that have executed successfully, and coupled to the allocator/renamer to communicate a de-allocate signal to the allocator/renamer. See Fig.7, Fig.8, and column 14, lines 12-20. Also, it is inherent that when an instruction is retired, its resources (i.e. registers) are deallocated so that another instruction has the option to use them (as opposed to resources continuing to be allocated to an instruction which no longer requires them because that instruction has completed). If these resources were not deallocated, then they would not be available to the processor, thereby inhibiting execution.

21. Referring to claim 17, Sager has taught a method comprising:

a) receiving an instruction of a plurality of instructions. See Fig.7 and column 9, lines 36-43 and note that the checker receives instructions from a delay unit.

b) placing the instruction in a queue with other instructions of the plurality of instructions. See Fig.7 and column 9, lines 50-53, and note that the checker places instructions in a queue (buffer). Recall that this buffer can hold multiple instructions. See column 9, line 63, to column 10, line 2.

c) speculatively re-ordering those of the plurality of instructions in a scheduler based on data dependencies and instruction latencies. See column 8, lines 52-63.

d) dispatching one of the plurality of instructions to an execution unit to be executed. See column 8, lines 64-67.

e) executing the instruction. See column 8, lines 64-67.

f) determining whether the instruction executed successfully. See Fig.7 and column 9, lines 31-36, and note the checker verifies the instruction's execution.

g) routing the instruction and all associated dependent instructions back to the queue if the instruction did not execute successfully. See column 9, lines 50-53. Also, see column 9, lines 31-36, column 10, lines 47-52, and column 12, lines 4-8. From these passages, Sager has made it clear that when an independent instruction needs to be replayed, its dependent instructions also need to be replayed.

h) retiring the instruction if the instruction executed successfully and allowing the instruction's associated dependent instructions to execute. See Fig.7, Fig.8, and column 14, lines 12-20. In addition, see column 10, lines 52-56, and column 12, lines 32-37. More specifically, if an independent instruction has executed unsuccessfully, then the data it has produced will be incorrect, causing itself and all dependent instructions to be replayed. However, if the independent instruction executes successfully, then it will have produced correct data and the dependent instructions will no longer need to be replayed, i.e., they can execute successfully (as long as they themselves are not incorrectly processed for some reason).

i) the instruction and associated dependent instructions are routed to the queue until data required for successful execution of the instruction is valid. See column 9, lines 31-36, column 10, lines 47-52, and column 12, lines 4-8. From these passages, Sager has made it clear that when an independent instruction needs to be replayed, its dependent instructions also need to be replayed.

22. Referring to claim 18, Sager has taught a method as described in claim 17. Sager has further taught allocating those of a plurality of system resources needed by the instruction. See column 13, lines 61-64.

23. Claims 1-4, 6, 9-15, and 17-19 rejected under 35 U.S.C. 102(e) as being anticipated by Merchant et al., U.S. Patent No. 6,212,626 (herein referred to as Merchant).

The applied reference has a common assignee with the instant application. Based upon the earlier effective U.S. filing date of the reference, it constitutes prior art under 35 U.S.C. 102(e). This rejection under 35 U.S.C. 102(e) might be overcome either by a showing under 37 CFR 1.132 that any invention disclosed but not claimed in the reference was derived from the inventor of this application and is thus not the invention "by another," or by an appropriate showing under 37 CFR 1.131.

24. Referring to claim 1, Merchant has taught a processor comprising:

a) a replay queue to receive a plurality of instructions. See the replay system 70 in Fig.1. More specifically, stages 84 and 85 would make up the replay queue.

b) an execution unit to execute the plurality of instructions. See Fig.1, component 58.

c) a scheduler coupled between the replay queue and the execution unit to speculatively schedule instructions for execution. See Fig.1 and note that the replay mux is a scheduler since it selects one of multiple instructions to send to the execution core. It should be seen that the output instruction of the replay queue goes to the mux, wherein the mux will eventually send that instruction to the execution unit.

d) a checker coupled to the execution unit to determine whether each instruction of the plurality of instructions has executed successfully, and coupled to the replay queue to dispatch to the replay queue each instruction that has not executed successfully. See Fig.1, component 72.

e) independent instructions and associated dependent instructions are moved to the replay queue upon unsuccessful execution of an independent instruction until data required for successful execution of the independent instruction is valid. See Fig.6A-6E and column 6, line 46, to column 7, line 10. More specifically, note that the instruction written in cycle 2 (Fig.6B) is dependent on the instruction written in cycle 1 (Fig.6A). This is denoted by the use of the letter "D" in Fig.6B. Likewise, the instruction written in cycle 3 (Fig.6C) is dependent on the instructions written in cycles 1 and 2 (Fig.6A-B). This is denoted by the use of two letter "D's" in Fig.6C. Also, in Fig.6C, it is determined that instruction 1 must be replayed (since it failed the check). As a result dependent instruction 2 and 3 must also be replayed, as shown in Fig.6D-E.

25. Referring to claim 2, Merchant has taught a processor as described in claim 1. Merchant has further taught an allocator/renamer coupled to the replay queue to allocate and rename those of a plurality of resources needed by the instruction. See Fig.1 and Fig.2 and note that the

scoreboard deals with renaming and allocation.

26. Referring to claim 3, Merchant has taught a processor as described in claim 2. Merchant has further taught a front end coupled to the allocator/renamer to provide the plurality of instructions to the allocator/renamer. See Fig.1, component 52.

27. Referring to claim 4, Merchant has taught a processor as described in claim 2. Merchant has further taught a retire unit to retire, the plurality of instructions, coupled to the checker to receive those of the plurality of instructions that have executed successfully, and coupled to the allocator/renamer to communicate a de-allocate signal to the allocator/renamer. See Fig.1, component 62. Also, when an instruction is retired, its resources (i.e. registers) are deallocated so that another instruction has the option to use them. See column 3, lines 27-29.

28. Referring to claim 6, Merchant has taught a processor as described in claim 1. Merchant has further taught:

a) at least one cache system on a die of the processor. Note the disclosure of a cache in column 3, lines 53-54. For a cache miss to occur, a cache system must exist.

b) a plurality of external memory devices. See column 2, lines 24-27. The existence of main memory and/or hard disk is inherent. These types of slower, but larger memories are used to hold programs and data for execution.

c) a memory request controller coupled to the execution unit to obtain a plurality of data from the at least one cache system and the plurality of external memory devices and to provide the plurality of data to the execution unit. See Fig.1, bus 98, and column 2, lines 24-27. Note that a controller would be required to retrieve and send data along bus 98 to some memory device.

29. Referring to claim 9, Merchant has taught a processor as described in claim 1. Merchant

has further taught a staging queue coupled between the checker and the scheduler. See Fig.1, components 80, 81, 82, and 83.

30. Referring to claim 10, Merchant has taught a processor as described in claim 1. Merchant has further taught that the checker comprises a scoreboard to maintain a status of a plurality of resources. See Fig.1, component 54, and Fig.2.

31. Referring to claim 11, Merchant has taught a processor comprising:

a) a replay queue to receive a plurality of instructions. See replay system 70 in Fig.1. More specifically stages 84 and 85 would make up the replay queue.

b) at least two execution units to execute the plurality of instructions. See Fig.1, component 58, and column 2, lines 65-66.

c) at least two schedulers coupled between the replay queue and the execution units to schedule instructions for execution based on data dependencies and instruction latencies. See the checker 72 and the replay mux 56 in Fig.1. Note that the mux is a scheduler in that it chooses between instructions supplied by the actual scheduler itself and instructions supplied by the replay queue. Also the checker schedules instructions to be replayed based on indications from the execution units.

d) a checker coupled to the execution units to determine whether each instruction has executed successfully, and coupled to the replay queue to communicate each instruction that has not executed successfully. See Fig.1, component 72.

e) independent instructions and associated dependent instructions are moved to the replay queue upon unsuccessful execution of an independent instruction until data required for successful execution of the independent instruction is valid. See Fig.6A-6E and column 6, line 46, to column 7, line 10. More specifically, note that the instruction written in cycle 2 (Fig.6B) is dependent on the instruction written in cycle 1 (Fig.6A). This is denoted by the use of the letter "D" in Fig.6B. Likewise, the instruction written in cycle 3 (Fig.6C) is dependent on the instructions written in cycles 1 and 2 (Fig.6A-B). This is denoted by the use of two letter "D's" in Fig.6C. Also, in Fig.6C, it is determined that instruction 1 must be replayed (since it failed the check). As a result dependent instruction 2 and 3 must also be replayed, as shown in Fig.6D-E.

32. Referring to claim 12, Merchant has taught a processor as described in claim 11. Merchant has further taught a plurality of memory devices coupled to the execution units such that the checker determines whether the instruction has executed successfully based on a plurality of information provided by the memory devices. See Fig.2 and note that the scoreboard is a memory device that allows the checker to determine whether an instruction's execution was successful. The scoreboard's operation per cycle is dependent on the instruction that has been executed, which has been ultimately provided by the instruction queue memory (Fig.1).

33. Referring to claim 13, Merchant has taught a processor as described in claim 12. Merchant has further taught an allocator/renamer coupled to the replay queue to allocate and rename those of a plurality of resources needed by the plurality of instructions. See Fig.1 and Fig.2 and note that the scoreboard deals with renaming and allocation.

34. Referring to claim 14, Merchant has taught a processor as described in claim 13. Merchant has further taught a front end coupled to the allocator/renamer to provide the plurality of instructions to the allocator/renamer. See Fig.1, component 52.

35. Referring to claim 15, Merchant has taught a processor as described in claim 13. Merchant has further taught a retire unit to retire the plurality of instructions, coupled to the

checker to receive those of the plurality of instructions that have executed successfully, and coupled to the allocator/renamer to communicate a de-allocate signal to the allocator/renamer. See Fig.1, component 62. Also, when an instruction is retired, its resources (i.e. registers) are deallocated so that another instruction has the option to use them. See column 3, lines 27-29.

36. Referring to claim 17, Merchant has taught a method comprising:

a) receiving an instruction of a plurality of instructions. See Fig.1 and note that the checker receives an instruction via staging queue 80-83.

b) placing the instruction in a queue with other instructions of the plurality of instructions. See Fig.1 and note that the instruction may be placed in replay queue 84-85.

c) speculatively re-ordering those of the plurality of instructions in a scheduler based on data dependencies and instruction latencies. See column 2, lines 15-17, and lines 38-53. Note that instructions cannot be executed until their resources are available and the availability of these resources is dependent on the latencies of the instructions producing those resources.

d) dispatching one of the plurality of instructions to an execution unit to be executed. See Fig.1 and column 2, lines 62-65.

e) executing the instruction. See Fig.1 and column 2, lines 62-66.

f) determining whether the instruction executed successfully. See column 3, lines 46-48.

g) routing the instruction and all associated dependent instructions back to the queue if the instruction did not execute successfully. See Fig.1 and column 3, lines 17-32. Also, see Fig.6A-6E and column 6, line 46, to column 7, line 10. More specifically, note that the instruction written in cycle 2 (Fig.6B) is dependent on the instruction written in cycle 1 (Fig.6A). This is denoted by the use of the letter "D" in Fig.6B. Likewise, the instruction written in cycle 3

(Fig.6C) is dependent on the instructions written in cycles 1 and 2 (Fig.6A-B). This is denoted by the use of two letter "D's" in Fig.6C. Also, in Fig.6C, it is determined that instruction 1 must be replayed (since it failed the check). As a result dependent instruction 2 and 3 must also be replayed, as shown in Fig.6D-E. And, replayed instructions are stored in a queue.

h) retiring the instruction if the instruction executed successfully and allowing the instruction's associated dependent instructions to execute. See Fig.1, component 62 and column 3, lines 23-27. Also, see Fig.6F-H, and column 7, lines 11-42. Note that the instruction is declared replay safe (i.e., it passed the check and has executed successfully). As a result, in Fig.6G-H, the instruction's dependents are also allowed to execute and are declared replay safe (replay is not needed).

i) the instruction and associated dependent instructions are routed to the queue until data required for successful execution of the independent instruction is valid. See Fig.6A-6E and column 6, line 46, to column 7, line 10. More specifically, note that the instruction written in cycle 2 (Fig.6B) is dependent on the instruction written in cycle 1 (Fig.6A). This is denoted by the use of the letter "D" in Fig.6B. Likewise, the instruction written in cycle 3 (Fig.6C) is dependent on the instructions written in cycles 1 and 2 (Fig.6A-B). This is denoted by the use of two letter "D's" in Fig.6C. Also, in Fig.6C, it is determined that instruction 1 must be replayed (since it failed the check). As a result dependent instruction 2 and 3 must also be replayed, as shown in Fig.6D-E.

37. Referring to claim 18, Merchant has taught a method as described in claim 17. Merchant has further taught allocating those of a plurality of system resources needed by the instruction. See the scoreboard in Fig.1 and column 2, lines 43-44.

38. Referring to claim 19, Merchant has taught a method as described in claim 18. Merchant has further taught that retiring comprises:

a) de-allocating those of the plurality of system resources used by the instruction being retired. See column 3, lines 17-29.

b) removing the instruction and a plurality of related data from the queue. If an instruction is eligible for retirement, then there is no need to keep it in the queue, since it won't need to execute again. Therefore, it is inherent that the instruction along with its data will be removed. Otherwise, if instructions were not removed, once the replay queue fills up due to its finite storage space, it will stay full and no additional instructions would be able to be stored for replay purposes.

Claim Rejections - 35 USC § 103

39. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

40. Claims 5, 16, and 19 are rejected under 35 U.S.C. 103(a) as being unpatentable over Sager, as applied above, in view of Baxter et al., U.S. Patent No. 5,944,818 (herein referred to as Baxter).

41. Referring to claim 5, Sager has taught a processor as described in claim 4. Sager has not explicitly taught that the retire unit is further coupled to the replay queue to communicate a retire signal when one of the plurality of instructions is retired such that the retired instruction and a

plurality of associated data are removed from the replay queue. However, Baxter has taught such a concept. See Fig.2 and column 3, lines 43-55, and note that upon retirement, the corresponding instruction entry in the replay queue (MIQ) is discarded (via deallocation signal shown in Fig.2) since there is no longer a need to maintain the instruction. Likewise, when an instruction retires in Sager, there would be no need to maintain that instruction in the replay queue. Doing so would consume resources for no beneficial reason. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Sager in view of Baxter so that upon retirement of an instruction, the retire unit communicates a signal to the replay queue in order to remove that instruction and its associated data (such as branch prediction information as disclosed in column 5, lines 64-66) from the queue.

42. Referring to claim 16, Sager has taught a processor as described in claim 15. Sager has not explicitly taught that the retire unit is further coupled to the replay queue to communicate a retire signal when one of the plurality of instructions is retired such that the retired instruction and a plurality of associated data are removed from the replay queue. However, Baxter has taught such a concept. See Fig.2 and column 3, lines 43-55, and note that upon retirement, the corresponding instruction entry in the replay queue (MIQ) is discarded (via deallocation signal shown in Fig.2) since there is no longer a need to maintain the instruction. Likewise, when an instruction retires in Sager, there would be no need to maintain that instruction in the replay queue. Doing so would consume resources for no beneficial reason. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Sager in view of Baxter so that upon retirement of an instruction, the retire unit communicates a signal to

the replay queue in order to remove that instruction and its associated data (such as branch prediction information as disclosed in column 5, lines 64-66) from the queue.

43. Referring to claim 19, Sager has taught a method as described in claim 18.

a) Sager has further taught that retiring comprises de-allocating those of the plurality of system resources used by the instruction being retired. It is inherent that when an instruction is retired, its resources (i.e. registers) are deallocated so that another instruction has the option to use them (as opposed to resources continuing to be allocated to an instruction which no longer requires them because that instruction has completed). If these resources were not deallocated, then they would not be available to the processor, thereby inhibiting execution.

b) Sager has not explicitly taught that retiring comprises removing the instruction and a plurality of related data from the queue. However, Baxter has taught such a concept. See Fig.2 and column 3, lines 43-55, and note that upon retirement, the corresponding instruction entry in the replay queue (MIQ) is discarded (via deallocation signal shown in Fig.2) since there is no longer a need to maintain the instruction. Likewise, when an instruction retires in Sager, there would be no need to maintain that instruction in the replay queue. Doing so would consume resources for no beneficial reason. As a result, it would have been obvious to one of ordinary skill in the art at the time of the invention to modify Sager in view of Baxter so that upon retirement of an instruction, the retire unit communicates a signal to the replay queue in order to remove that instruction and its associated data (such as branch prediction information as disclosed in column 5, lines 64-66) from the queue.

Response to Arguments

44. Applicant's arguments filed on February 24, 2004, have been fully considered but they are not persuasive.

45. In the remarks, Applicant, in general, argues the novelty/rejection of claims 1, 11, and 17, in substance that the references do not teach, disclose, or suggest correct and optimum replay scheduling of dependent instructions.

46. These arguments are not found persuasive for the following reasons:

a) Both Sager and Merchant has taught replay scheduling of dependent instructions. This has been shown in the rejections above. In addition, a person of ordinary skill in the art would expect such replay scheduling of dependents to occur. For example, take the following instruction sequence:

```
ADD R1, R2, R3      //R1 = R2 + R3      (instruction #1)
SUB R4, R5, R1      //R4 = R5 - R1      (dependent on instruction #1)
```

Since the SUB instruction uses R1 in a subtraction operation and the ADD instruction establishes the value in R1, the SUB instruction must wait for the ADD instruction to finish. Therefore, if the ADD instruction executes successfully, R1's value will be set successfully, and the SUB instruction will be able to proceed. However, if the ADD instruction executes unsuccessfully, then R1 will either not be set at all or set to the wrong value. Either way, the SUB instruction, if allowed to execute, will perform the subtraction with the wrong data. Consequently, it should be realized that if the ADD needs to be replayed, then the SUB will need to be replayed as well so that the correct value of R1 will be used in the subtraction operation.

Conclusion

47. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to David J. Huisman whose telephone number is (703) 305-7811. The examiner can normally be reached on Monday-Friday (8:00-4:30).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (703) 305-9712. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

DJH
David J. Huisman
March 4, 2004

[illegible]

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100